检测方法
通过结合直接和间接的三维带指数时间衰退策略的信任度（满意度、适应性、连接程度），以检测自私/恶意节点并转发消息。由于在自私节点比例较大的场景下，仅依赖最终ACK反馈不能获得足够的信息，该方法提出两跳ACK反馈的策略。
在计算源自信任推荐的间接信任度时，该方法基于推荐节点本身的信任度和相似度评价进行筛选和更新。除了普通的单个节点的黑洞/灰洞自私行为（丢弃攻击）以外，自私节点还可能通过制造虚假的消息投递记录来误导检测系统。

A Dynamic Trust Framework for Opportunistic Mobile Social Networks

Eric Ke Wang, Yueping Li, Yunming Ye, S. M. Yiu, *Member, IEEE*, and Lucas C. K. Hui, *Senior Member, IEEE*

***Abstract*—Opportunistic mobile social network (OMSN) enables users to form an instant social network for information sharing (e.g., people watching the same soccer game can share their instant comments). OMSN is ad hoc in nature, thus relies on the cooperation of members regarding message transmission. However, some uncooperative or malicious behavior from abnormal members may reduce network performance, even damage the entire network. Currently, there does not exist effective mechanisms to detect selfish and malicious nodes. To tackle this problem, we propose a dynamic trust framework to facilitate a node to derive a trust value of another node based on the behavior of the latter. The novelty of our framework includes the following: 1) we design a new metric for a trust value of a node and 2) we propose a "two–hop feedback method" that requires intermediate nodes in a forwarding path to generate ACK messages to verify a node's honesty if they are two hops away. In most existing trust models, final ACK messages are considered as critical factors. In OMSN, nodes are not fully connected and final ACK messages cannot be reliably received. In order to avoid the problem that few final ACK messages can be received, we propose a "two–hop feedback method." Simulation results show that our approach is able to detect a majority of abnormal nodes including malicious nodes, selfish nodes, and those nodes launching conspiracy attacks. Thus, the entire network efficiency can be improved without negative impact of abnormal nodes. Besides, our trust framework can be easily applied to the current popular routing protocols of opportunistic networks.**

***Index Terms*—Distributed management, ad-hoc and sensor networks, self-management and autonomic networks, security management, distributed platforms.**

Manuscript received January 27, 2017; revised June 20, 2017, October 2, 2017, and November 15, 2017; accepted November 19, 2017. Date of publication November 22, 2017; date of current version March 9, 2018. This research was supported in part by National Natural Science Foundation of China (No.61572157), grant No.2016A030313660 and 2017A030313365 from Guangdong Province Natural Science Foundation, grants JCYJ20160608161351559, JCYJ20150617155357681, JCYJ20160428092427867, JSGG20150512145714247 from Shenzhen Municipal Science and Technology Innovation Project. The associate editor coordinating the review of this paper and approving it for publication was C. Fung. *(Corresponding author: Yueping Li.)*

E. K. Wang is with the Department of Computer Science, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: wk_hit@hit.edu.cn).

Y. Li is with the School of Computer Engineering, Shenzhen Polytechnic, Shenzhen 518055, China (e-mail: leeyueping@gmail.com).

Y. Ye is with the Key Laboratory of Shenzhen Internet Information Collaborative Technology and Application at Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: yym@hitsz.edu.cn).

S. M. Yiu and L. C. K. Hui are with the Department of Computer Science, University of Hong Kong, Hong Kong (e-mail: smyiu@cs.hku.hk; hui@cs.hku.hk).

Digital Object Identifier 10.1109/TNSM.2017.2776350

I. Introduction

With the tremendous increase of smart phone users, a new type of social networks, called *opportunistic mobile social network (OMSN)*, has emerged recently. It is different from a traditional social network, which has a centralized host. OMSN is a distributed peer-to-peer mobile ad hoc network. There is no guarantee of a path from a source to a destination due to the highly dynamic nature of the network. In an OMSN, users with mobile devices are able to form an on-the-fly social network to communicate with one another. The communication channels are instant and ephemeral. It allows people to achieve instant sharing within some social life activities. For example, when watching a big sport game or show in a stadium, people can form an instant BBS over Bluetooth to chat and share information for the game. Some popular applications of OMSN include the photo instant sharing application: Snapchat [1], Opportunistic MSN [2]. In this paper, we focus on the networks formed by humans carrying personal mobile devices during their daily lives.

OMSN requires cooperation among members. It relies on the willingness of members to forward and relay messages. However, mobile devices usually have limited memory, low power, and are expensive for communications. Without a centralized host, some members may behave uncooperatively by refusing to forward messages or spreading large amount of useless messages, that lead to increase network load or decrease the performance of network. Besides traditional malicious members, there are some *selfish* members that only receive messages, but refuse to forward messages. Even for malicious members, the attack models are different from traditional networks (e.g., black hole attack [3], epidemic attack [4] and conspiracy attack [5]). While it is difficult to prohibit those selfish and malicious nodes from joining network due to its open nature, handling them in network to improve safety and throughput becomes important.

Intuitively, if those problematic nodes can be detected and other normal nodes can avoid or minimize to respond with them, it is able to reduce the negative impact on network substantially. In this paper, we propose a new trust framework to build a trust based routing mechanism for OMSN. Instead of using cryptographic techniques, we tackle it by a dynamic trust model. Technically, we derive a new trust quantification method based on the combination of three factors: *satisfaction*, *fitness* and *connectivity* (See Section III-B1 for more details). When quantifying the trust of a node, most existing methods rely on the number of final ACK messages received by the node. However, in OMSN, final ACK messages cannot

be reliably received. To solve the problem that few final ACK messages are received, we propose a novel "two-hop feedback method" that requires two hops away nodes in a forwarding path to generate ACK messages when they receive messages, instead of only relying on final ACK messages generated from the destination. The proposed trust model is applicable to existing popular routing algorithms such as Prophet, SprayWait and Epidemic for OMSN.

II. Related Work

Traditional trust models have been studied for a long time. Beth *et al.* [6] and Yahalom *et al.* [7] described some of the earliest research on trust models. They proposed to compute the trust of a node by calculating the arithmetic mean of the recommended trust values provided by other nodes. In their model, they classify trust into *direct* trust and *indirect* trust that have been well cited and acknowledged by researchers. After that, Yu and Singh [8] and Venkatraman *et al.* [9] proposed a social network based trust model to compute the trust values of target nodes. The model requires a stable social network, so it is not applicable to OMSN. The concept of social network has also been employed in the trust models of Caverlee *et al.* [10], [11]. Their ideas are to use community discovery algorithms to identify whether a node is inside a community or not, they still relies on stable and quantifiable social network.

Some researchers have proposed distributed trust models. Xiong and Liu [12] proposed Peertrust, that mainly computes the trust value of target nodes based on others' feedback, however, it depends too much on the feedback from other nodes while feedback may not be in time and it also may suffer from conspiracy attacks (collusion between members). Kamvar *et al.* [13] presented EigenTrust, that relies on a *weighted trust transference method* to compute the trust value of target nodes, but it needs to pre-set the trusts of some nodes, which makes it hard for OMSN. Zhou and Hwang [14] gave a Powertrust to overcome the problem of pre-setting the trusts of some nodes by identifying an initial set of trustworthy nodes based on past behaviors of the nodes, however, it is easy for malicious nodes to cover their malicious behaviour by performing multiple normal operations. Josang and Haller [15] provided a distributed trust model Dirichlet, that mainly employs Dirichlet Probability distribution theory to compute trust, the computation process is relatively simple and easy to realize, but it lacks the defense techniques against malicious nodes that launch conspiracy attacks. Chun-Qi *et al.* [16] considered the dynamic behavior of nodes that may affect the trust computation and proposed R2BTM by introducing the risk parameter to identify the nodes with abnormal behaviour, but R2BTM does not provide a quantitive measure. FCTrust [17] is a distributed P2P overall trust model based on feedback from other nodes, it assigns the nodes providing feedback different weights that make the computation more accurate than other feedback based model. Li *et al.* [18] proposed an encounter ticket scheme that counts the number of encounter tickets collected and uses a probability model to evaluate trust values. The scheme may suffer from the tailgating attack of some malicious nodes that can exaggerate trust values. Besides, the risk of forging and tampering encounter tickets, that are transmitted over the network, has not been analyzed formally.

The above trust models are mainly designed for traditional networks (with a centralized host) or P2P networks. OMSN is highly dynamic, ad hoc and open, it poses new challenges. More recent works include the following, but none of them addressed OMSN. Liu and Wang [19] proposed a trust control scheme for heterogeneous networks of Internet of Things. It mainly solves the overall trustiness across heterogeneous networks. Saied *et al.* [20] presented a context-aware and multi-service trust management scheme, but it requires a centralized trust manager which cannot be applied to a completely distributed model. Chen *et al.* [21] and Mahalle *et al.* [22] gave two trust management models based on fuzzy approaches, one is fuzzy reputation, the other is fuzzy based access control, that mainly employ fuzzy approach to compute the trustiness value, however, it can not be applied to OMSN. Nitti *et al.* [23] introduced a trustiness management scheme for social Internet of Things which assumes that the interaction of devices is similar to the interaction of humans. It gives devices mimic society features, such as two devices produced by a same manufacturer have certain natural relationship. It is interesting to introduce social features. However, the effectiveness in OMSN is not discussed.

To summarize,although trust model has been studied for a long time [24], [25], those trust models were not designed for OMSN. In this paper, we mainly propose an effective trust framework for OMSN.

III. Our Proposed Trust Framework

A. Architecture of Trust Framework

Our proposed trust framework consists of four following parts, detection of behaviour, delivery of trust, processing of trust and decision of trust as shown in Figure 1. Detection of behaviour is to capture the bad behaviour of abnormal nodes by recording the number of times of forwarding, non-cooperation, and successful message delivery. Delivery and storage of trust degree involve the storage of local direct trust and indirect trust degrees, selective acceptance of neighbour node's recommendation of other nodes' trust degree. Trust degree is defined as a vector, composed of three dimensions: connectivity, fitness, and satisfaction (Described in Section III-B1). Processing of trust is to perform aggregated calculation by using both local trust degree and recommended trust degree, with a time decay strategy.

B. Trust Model

We first describe two related works in details. A Beyesian trust algorithm for self-organizing networks was proposed by Denko *et al.* [26]. It is a classical trust algorithm, which estimates the trust degree of a node by counting the number of successful and unsuccessful deliveries of messages. In the algorithm, each node has all the other nodes' records of their communication history. Each node collects the other nodes' recommendations. Nodes decide whether to accept a

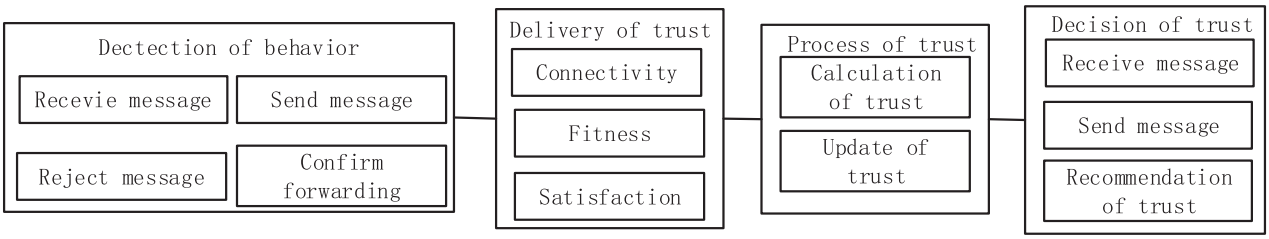


Fig. 1. Trust model architecture.

recommendation by considering the referee's reliability of recommendation, so that it can avoid a lot of unnecessary computation. Besides, failure history is recorded, so that it can recognize those uncooperative nodes easily. This message would be broadcasted in the network, thus normal nodes would avoid communicating with those uncooperative nodes or help forwarding their messages. This strategy can effectively motivate more cooperation desire of nodes. However, it depends on the collection of the final ACK messages sent from the target nodes. Once the final ACK messages are received, nodes mark who have been helpful for their messages, successful transmission rate would increase later on. Therefore there could be another situation. If the network has many selfish nodes, it has a high chance that selfish nodes drop their received messages when they act as intermediate nodes. thus, source nodes never receive the final ACK messages from destination nodes, and can not estimate the trust degrees of intermediate nodes accurately. Besides, detection of conspiracy attacks among malicious nodes is also not taken into consideration.

Our improvement on Bayesian algorithm is inspired by an idea proposed by Chen *et al.* [27]. They proposed a trust quantification method and defined trust in four dimensions {unselfishness, healthness, connectivity, energy} and in two types, QoS trust and Social trust. They adopted normal weighted average method of computing four dimensions, direct trust and indirect trust. But there are some problems in their approach. For example, indirect recommendation should be trusted in different levels, which depends on a recommender's confidence and reliability. So the direct trust should also be computed based on some previous experience. We think that Bayesian approach is a good solution to solve this problem. So combining Bayesian algorithm with a suitable trust quantification method is a possible solution. Based on the features of OMSN, we design and propose a more adaptive and efficient model for OMSN.

Our trust model has a new quantification method that uses 3 dimensions (instead of 4 dimensions when compared to Chen *et al.*'s approach): connectivity, fitness, satisfaction. We try to tackle the deficiency that our Bayesian trust algorithm has fewer trust attributes (since if there are lots of selfish or malicious nodes in the network, final ACK messages will not be transmitted by non-cooperative nodes and Bayesian trust algorithm cannot run well without enough final ACK messages). In calculating a trust degree, we use a "two-hop feedback method" that requires two hops away intermediate nodes to generate ACK messages for the nodes. Thus, senders do not need to wait for receiving final ACK messages which may never arrive. The flow of our model is shown in Figure 2.

1) Quantification of Direct Trust Value: In OMSN, three abnormal behaviours, selfishness, conspiracy and flooding, occur more frequently than other abnormal behaviours. Traditional trust model only based on probability can not effectively reveal the above three abnormal behaviors. Therefore, we describe the trust of nodes by using vectors of three dimensions. A trust vector is composed of connectivity, fitness and satisfaction. Each dimension represents a trust degree in its own aspect, the value ranges from 0 to 1. Higher value represents higher trust in its dimension. We define the formulas in three dimensions.

A.Connectivity Connectivity of nodes is the ability of a node to connect another node in the network. In OMSN, connectivity is unpredictable. But the frequency of connection between two nodes can be counted. In order to enhance the accuracy of connectivity, nodes observe all forwarding paths and consider the number of times that a node is in a forwarding path as a factor. Normal nodes in the network will run actively to get high connectivity. But some nodes will run in a low speed to reduce cost of energy. The formula for calculating connectivity is defined as (1):

$$T_{i,j}^{d,c} = \left(2 * n_{fwd} + n_{meet}\right) / \left(2 * n_{fwd} + n_{meet} + N\right) \quad (1)$$

$T_{i,j}^{d,c}$ represents the direct connectivity from node i to node j, n_{fwd} is the number of times that node j is an intermediate node in a forwarding path. n_{meet} is the number of times that node i and node j are connected. N is the total number of nodes in the network.

B.Fitness Fitness of a node is a standard to judge the possibility of a node to be a normal node. Its purpose is to detect the flooding behaviour or blackhole attacks. Each message carries its own forward path, which includes source nodes and intermediate forwarding nodes. The fitness of a node is based on forwarding paths and direct interaction records. It is mainly decided by the count of the messages sent by source nodes, number of received messages, number of denial and forwarding messages. Formula 2 shows how fitness is computed:

$$T_{i,j}^{d,h} = \left(n_{fwd} + n_{rec} + 1\right) / \left(n_{src} + n_{fwd} + n_{rec} + n_{deny} + 2\right) \quad (2)$$

$T_{i,j}^{d,h}$ represents the fitness that node i calculate for node j according to the interaction records. n_{fwd} is the number of messages node j forwarded. n_{rec} is the count of node j receiving node i's messages. n_{src} is number of messages sent by node j as a source node. n_{deny} is the count for the denial of messages.

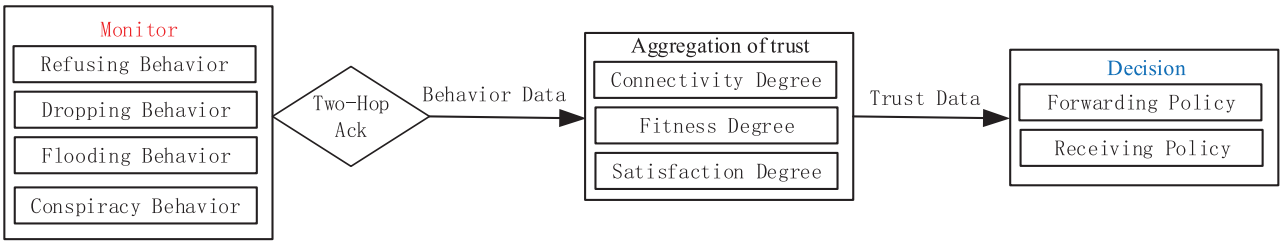


Fig. 2. The Flow of Trust Evaluation.

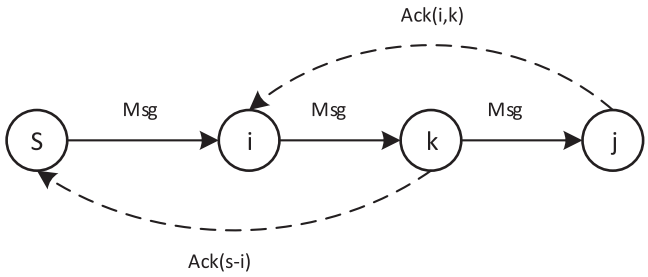


Fig. 3. Process of Ack message delivery.

Ack ID	Msg ID	Target node	Source node	Forward path	Generation time	Survival time

Fig. 4. Format of final Ack.

C.Satisfaction Satisfaction is the degree of how a node is satisfied with the forwarding behaviour done by intermediate nodes. It is decided by the number of received ACK messages and the number of forwarding messages. Nodes wait for feedback messages, after forwarding message to the next-hop node. When nodes receive the valid ACK messages, they confirm the success of the forwarding message by the intermediate nodes and record the count. Otherwise, if the ACK messages cannot be received within a certain length of time, the forwarding is considered failed and the count will not be increased.

D.2 – Hop Ack

In this section, we describe our *2-Hop ACK* mechanism. The 2-Hop ACK mechanism is shown in Figure 3.

In Figure 3, node i successfully receives and stores Msg from storage node S, and move in the network with Msg carried. At some time, node i meets neighbour node k. First, i will judge if k can be trusted. If yes, node i will forward message to node k. Upon receiving Msg from i, Node k will judge whether the length of the forwarding path is at most 2. If yes, node k will generate ACK response of Msg. This ACK's target node is S, forwarding node is i, source node is k. The time of receiving Msg is defined as the generation time of k. ACK is also assigned with survival time, which is used to be short. Forwarding nodes and MsgID can mark ACK message. This model can use single or flooding to forward ACK message. In limited time, if node S receives this ACK message, node S would pose a positive attitude by adding 1 to feedback message counter. Thus, after each node send message to next node, they would monitor feedback message ACK sent to itself in the network. Upon receiving valid ACK, forwarding nodes recorded in ACK will be marked with positive record. The format of ACK is shown in Figure 4.

In a final ACK message, the forwarding path of Msg and the time it resides in each node are included. Node S does ACK confirmation calculation on each node in the path, and use the count of ACK messages to get the cooperation degree of the next node. Final ACK message is more important, the survival time is longer.

For example, node A passes Msg to node B, node B passes Msg to node C, C sends ACK to A upon receiving Msg. Then, node A can confirm the good behaviour of node B according to the ACK message. After Msg was delivered to the target node D, D sends a final ACK to source node A. After receiving the final ACK, A confirms the good behaviour of all nodes in the forwarding path.

The calculating of satisfaction is shown in Formula (3):

$$T_{i,j}^{d,s} = \begin{cases} \frac{n_{ack}}{n_{rec}+1} & \text{if } path\ (j) = 0 \\ \frac{n_{Fack,j}+n_{ack}}{n_{Fack,j}+n_{rec}+1} & \text{if } path\ (j) > 0 \end{cases} \quad (3)$$

$T_{i,j}^{d,s}$ represents node i's direct satisfaction of node j, n_{ack} is the number of received ACK messages, n_{rec} is the number of times node j receives messages, $n_{Fack,j}$ is the number of occurrences of node j in final ACK messages, *path(j)* represents whether node j occurs in the path. In particular, when a node receives a final ACK message from a target node, it will confirm the good behaviour of each node in the forwarding path.

C. Delivery of Trust Value

In our framework, delivery of trust degree is designed to allow a node to know more comprehensively about other nodes' behaviour. By collecting recommendation of other nodes, current node can calculate a trust vector for other nodes more accurately, which makes up the deficiency in subjective judgment on trust. In order to reduce the malicious and fake recommendation and increase the reliability of recommendation, nodes will selectively accept recommendation of other nodes.

We define recommenders in two categories. One consists of the nodes with trust degree bigger than a certain threshold; the other contains the nodes that are very similar to the current node, which means that similar nodes behave similarly. Choosing nodes by threshold is a regular method, the latter one is for solving the case if the current node has none or very few interactions with other nodes. In the second method, if two nodes have high similarity in their blacklists(it

ACK怎么传播呢？flooding过去吗？

1.我和它trust高 所以信任它给出的推荐（高于阈值）
2.我与它的blacklist相似 所以信任它

两个node的blacklist 相似性比较高；即 他们 对 恶意节点的鉴别一致；推荐recommendation 应当被考虑接受

reflects their common sense on malicious nodes), then recommendation can be accepted. For example, node i makes the decision whether to accept neighbors' recommendation, two requirements should both be met: (i) the trust degree must be bigger than the threshold; and (ii) the node should have high similarity of their blacklists.

Since the computational power of each node is limited and the time on the path between two nodes is short, thus the measure method cannot be complex. We adopt Formula (4) to measure the similarity between blacklist of node i and node j:

怎样被认为应该加入blacklist ？

$$Sim(i,j) = \frac{|B_i \cap B_j|}{|B_i \cup B_j|} \tag{4}$$

B_i and B_j, respectively, represent the blacklist set of node i and node j, $|B_i \cup B_j|$ represents union of blacklist sets of node i and node j, $|B_i \cap B_j|$ represents intersection of blacklist set of node i and node j. Indirect trust can be calculated by Formula (5):

$$T_{i,m}^{ind,X}(t) = \begin{cases} \frac{\sum_{k \in R_i} \left\{ T_{i,j}^{d}(t) \times T_{j,m}^{d,X}(t) \right\}}{\sum_{j \in R_i} T_{i,j}^{X}(t)} & \text{if } T_{i,j}^{X}(t) > \tau \\ \frac{\sum_{j \in R_i} \left\{ Sim(i,j) \times T_{j,m}^{d,X}(t) \right\}}{\sum_{j \in R_i} Sim(i,j)} & \text{if } T_{i,j}^{X}(t) \le \tau \ Sim(i,j) > \nu \end{cases} \tag{5}$$

X represents the attributes corresponding to the trust, $T_{i,j}^{d}(t)$ represents node i's trust degree node j at time t. $Sim(i,j)$ represents the similarity between the blacklists of node i and node j, τ represents the threshold for the trust degree, ν represents the threshold of the similarity.

D. Aggregation and Update of Trust

Update of trust is divided into the update of direct and indirect trust. There are two situations in direct update. One is when node i has interaction with node j in period $|t, t+\Delta t|$. The other is when node i has no interaction with node j in period $|t, t+\Delta t|$. The calculation of the update of connectivity, fitness, satisfaction are shown in (6).

$$T_{i,j}^{d,X}(t+\Delta t) = \begin{cases} e^{-\lambda \Delta t} \times T_{i,j}^{d,X}(t) \\ \alpha \times e^{-\lambda \Delta t} \times T_{i,j}^{d,X}(t) + (1-\alpha) \times T_{i,j}^{d,X}(t+\Delta t) \end{cases} \tag{6}$$

For indirect trust, there are also two cases. One situation is that there is no received recommendation for node j during the period $|t, t+\Delta t|$, the other one is that there is no indirect recommendation received.

1.没有接收到对j的推荐
2.没有接收到任何推荐

$$T_{i,j}^{ind,X}(t+\Delta t) = \begin{cases} e^{-\delta \Delta t} \times T_{i,j}^{ind,X}(t) \\ \beta \times e^{-\delta \Delta t} \times T_{i,j}^{ind,X}(t) + (1-\beta) \times T_{i,j}^{ind,X}(t+\Delta t) \end{cases} \tag{7}$$

Aggregated trust degree is a trust value calculated with direct and indirect trust degree by a proposed strategy. For each attribute X, the trust from i to j at time $t+\Delta t$ is as follows:

$$T_{i,j}^{X}(t+\Delta t) = \gamma T_{i,j}^{d,X}(t+\Delta t) + (1-\gamma) T_{i,j}^{ind,X}(t+\Delta t) \tag{8}$$

Algorithm 1 Updating Trust Attribute Based on 2-Hop ACK

```
1)  getConnections(A); //acquire connection to neighbor A
2)  For each con in cons do: //For each connquit
3)     CalculateTrust(con); //calculate the trust
4)     If(IsTrust(con)):
5)        ReceiveMsg(Msg,con); //receiving message
6)        Add the value of rec;
7)        CreateNewMsg(AckId.Msg.carryNode);//generate feedback message
8)        Wait(TTL); //wait forwarding
9)        SendMsgToCon(otherMsg,con); //forward message
10)       Add the value of send;
11)       Listen(ACK); //wait for feedback message
12)       If(ACK.MsgId=Msg.Id): // nodes with ACK msg
13)             Add the Value of ACK(A,con);
```

According to the real condition of the network, γ is assigned with various values. If the situation of current network is more similar to direct trust, γ is assigned with bigger weight value. After the network runs for some time, the network tends to be more like indirect trust. Node i's trust to node j's forwarding capability can be represented by:

$$T_{i,j}(t+\Delta t) = \sum_{X}^{all} \omega^{X} T_{i,j}^{X}(t+\Delta t) \tag{9}$$

ω^X is the ratio of attribute X in all attributes. X is composed of satisfaction, connectivity and fitness.

To evaluate the satisfaction of nodes' cooperation, we used two feedback units 2-hop ACK and final ACK in the simulator. The former is employed for counting forwarding messages of active nodes, the latter is used to record feedback numbers of successful message delivery, enhancing the satisfaction of service in forwarding paths. On receiving normal messages, a node first judges whether the messages are ACK. If the messages are not ACK, it checks if intermediate forwarding nodes are involved in the forwarding paths. If yes, the corresponding ACK messages are generated. Corresponding information of forwarding nodes is carried in a small ACK message, sent to the last node. All nodes we designed here will forward this message without condition. The details are given in Algorithm 1.

A node carries message set M at time t. It collects nodes nearby to a set U. Then it checks for each node to see if it is reliable. If yes, it will be constructed to a node pair "*con*", and then be put into set "*cons*". Nodes in set "*cons*" will be sorted by their trust degrees. Messages sent by nodes in set M will be delivered to nodes around with higher trust degree. When a node gets a request for sending messages, it will first check the fitness of the request node. If the fitness is bigger than a threshold, it will determine that if the request node is in the state of delivering message and with enough memory. If both are met, it will receive the message.

E. Trust Decision

Trust decision are composed of four parts as follows:

1) *Decision of receiving messages:* If the fitness of the request node is less than a certain threshold, the current node would refuse the request for the message

forwarding. A node with low fitness has higher probability to have bad behaviour like denial of service or sending spam messages. Refusing message forwarding requests can motivate those nodes to behave better and gain a higher fitness value. If a node has low fitness such as it is a flooding node, refusing forwarding for that node can avoid garbage spreading in the network, which reduce the load of network and enhance the chance of forwarding normal messages.

2) *Decision of sending messages:* Nodes select those nodes with high fitness and connectivity to forward messages. Satisfaction is decided by degrees of services. If the satisfaction is higher, the rate of successful forwarding is higher, the probability of losing packets is lower. Nodes tend to select those nodes with higher satisfaction to forward messages, at the same time their connectivity would also be taken into consideration. Thus, rate of successful delivery would be improved under guarantee of high security.
3) *Decision of accept trust recommendation:* Whether nodes accept the trust recommendation depends on their trust threshold and blacklist similarity. Thus it can guarantee the authenticity and effectiveness of recommendation.
4) *Decision of blacklist:* If the fitness or satisfaction of some nodes is lower than a corresponding threshold, then they would be listed in blacklists and these blacklists would be broadcasted to neighbours.

IV. Simulation

We conducted a simulation of our framework in the Opportunistic Network Environment simulator (ONE) [28]. The simulation environment in ONE is a wireless network environment covering area of 4500m×3400m within a default Helsinki city map. The experiment settings including scenario setting, interface setting, node setting and event setting are shown in Tables I–V. In the experiment, various of selfish nodes and malicious nodes are produced to analyze their impact on the routing performance. We mainly test the performance of enforcing trust model on Epidemic, Prohet, SprayWait and FirstContact to tackle abnormal nodes. In Epidemic routing, nodes continuously replicate messages to newly arrived nodes that do not already have the message copy until predefined hop count's maximum value is reached. In Prohet routing, nodes employ probabilistic metric called delivery predictability to transfer messages to a reliable node. In SprayWait routing, it consists of two phases: spray phase and wait phase. In the spray phase, the source node initially spray L number of message copies to L distinct relay nodes. After receiving the message copy, all L, that relay nodes, go into the wait phase and wait until the transmission to the destination is done. In First Contact routing, each node forwards a package to the one with connection randomly. If there are no nodes to connect, the node would store the data until the emergence of connection. From the above four popular routers, Epidemic and SprayWait belongs to flooding families, while Prohet and FirstContact routers belong to forwarding families.

TABLE I
Setting of Experiments

Hardware	CPU: Intel(R) Corei5-3470@3.2GHz Memory: 16GB Second Cache: 6MB
Software	Operating System: Windows 7 64bit Developing Environment: Eclipse(Java)

TABLE II
Scenario Setting

Parameter	Hint	Value
sceneryName	Scenario Name	adaptive
endTime	Simulation Time	43200s
updateInterval	Update Period	0.1s
nrofHostGroups	Number of Hosts	2
simulateConnections	Connection Simulation	true

TABLE III
Interface Setting

Parameter	Hint	Value
btInterface.type	Type of Interface	SimpleBroadcast
btInterface.transmitRange	Range of Transimit	10m
btInterface.transmitSpeed	Speed of Transimt	250kbps

TABLE IV
Node Setting

Parameter	Hint	Value
Group.nrofHost	Number of Computers in Group	100
Movement	Movement Mode	ShortestMap
Group.waitTime	Waiting Time	0, 120s
Group.speed	Speed	0.5, 1.5m/s
Group.bufferSize	Cache	10M
Group.msgTTL	Lifetime of Message	5h

TABLE V
Event Setting

Parameter	Hint	Value
Events.nrof	Number of Event Generators	1
Events.interval	Message Generation Period	25, 35s
Events.size	Event Message Size	500K, 1M
Events.host	Host of Generators	0, 99
Event.tohost	Range of Destination	0, 99
Event.prefix	Prefix of Events	m

There are two kinds of mobile nodes in the city which are pedestrian and cars, and they communicate through Bluetooth equipment when they meet.

In the simulation, we set three types of nodes. The first group has nodes with their names start with "self", that drop every message with the name starts with "m". The second group has nodes with their names start with "deny", who deny any message except the care if the destination of the message is itself. The third group has nodes with their names start with "bad". These nodes will send out spam messages.

Based on the simulation, we realize selfish scenario, spam scenario and conspiracy attacks.

A. Experimental Results

In this section, we first learn the impact of abnormal nodes on the performance of the network. Second, we compare two

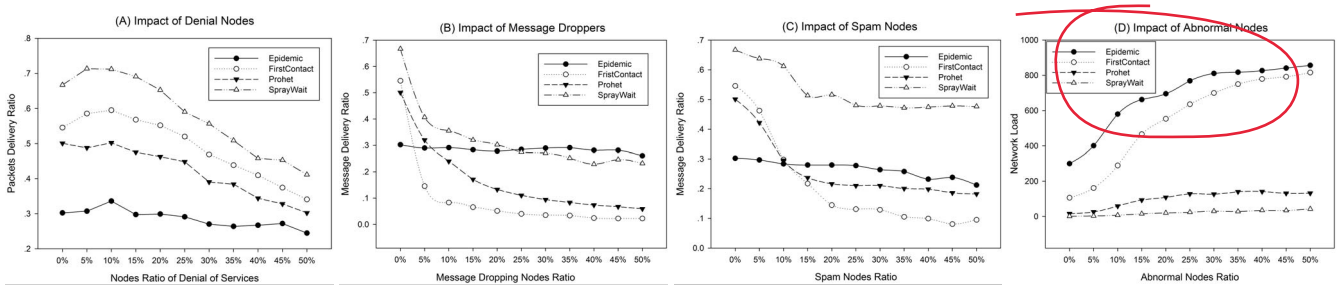


Fig. 5. Effect of different nodes.

scenes on the performance of different routers in simulation. One is with our trust model, the other is without any trust models. In the first scene, normal nodes and selfish nodes (denial of service, packet loss and no packet sending) are included. In the second scene, it involves a few groups of normal nodes and some abnormal nodes (spam behaviour). To evaluate our trust model, we base on four criteria:

1) *Rate of successful message delivery:* The percentage of those successful delivered messages in all messages sent by normal nodes. Messages sent by abnormal nodes are not included.
2) *Number of lost packets:* The number of packets that sent by normal nodes but dropped by other nodes.
3) *Load of network:* It is defined as the ratio of the number of messages that cannot reach the destinations over the number of successfully arrived messages.
4) *Detection rate of abnormal nodes:* Nodes count their own trust history, and compare it with nodes with low trust degree and abnormal nodes. Then, it will calculate the average precision rate of detection and recall rate.

1) Abnormal Behavior's Impact on Routing Protocols: In this section, we study the impact of abnormal nodes on the performance of the network. We injected different number of nodes with denial of service, packet of loss, and spam behaviour into the network. In our experiment, routing protocols: Epidemic, SprayWait, Prophet and FirstContact, were simulated.

(1) The effect of different numbers of denial nodes to routers is shown in Figure 5(A). The simulation shows that, in the environment with fewer selfish nodes, the rate of successful delivery increases by a small amount, which is caused by the feature that the nodes can move freely. The number of forwarding nodes in network should provide a better indicator. If the number of forwarding nodes is too high, the performance of spreading messages would decrease. The rate of successful delivery decreases as the number of denial nodes becomes more. The reason is that, with the increase of denial nodes, nodes cannot use the assistance of forwarding, most of them interact with each other directly. If all nodes are denial nodes, this router will be degenerated to Direct Delivery, which means that messages delivery can only be successful by direct communication without any forwarding.

(2) Nodes that lose packets make a serious effect to network, because packet loss will lead to permanently lost of messages, which is irreversible. Effect of different number of nodes with packet loss behaviour are shown in Figure 5(B).

Simulation result shows that these nodes have a much higher impact on the performance of the network, especially in router of single-copy or less-copies. FirstContact is a typical single-copy forwarding scheme, in which the nodes send messages to its first connected node. In this situation, it has a high chance for packet loss nodes to receive normal messages and drop the messages. Prophet and SprayWait have less copies of messages, thus packet loss nodes have an obvious affect on successful delivery in these two routers.

(3) Effect of different number of flooding nodes to routers is shown in Figure 5(C).

We set the speed of sending out messages for flooding nodes about 2∼3 times faster than that of normal nodes. The result shows that, the impact (to all four routers) on the performance of the network become greater with the increase of the percentage of flooding nodes. The performance is presented by the changing of the successful delivery with four scenes of four routers.

2) Selfish Nodes: In this experiment, we set several different percentages of selfish nodes, which is used to drop the received messages. After a trust model is added, normal nodes choose to send messages to nodes with high trust degree, which can enhance the rate of successful delivery and reduce the number of dropping messages.

(1) Successful delivery of messages

Experiment result shows that, in the original algorithm, rate of successful message delivery decreases as the number of selfish nodes increases. After trust model is added, rate of successful delivery is improved obviously. As the number of selfish nodes increases, the rate of successful delivery decreases slowly. The trust model improves all four routers' performance with different degrees, especially for single-copy router or router with small number of copies.

(1.1) Successful delivery rate under Epidemic router

Epidemic is an algorithm based on multi-copy flooding strategy, packet loss behaviour of selfish nodes has little affect to the delivery of normal messages. Because there are too many copies of messages in the network, the trust model in improved Epidemic cannot identify the selfish behaviour of nodes, which means that the trust model does not make obvious improvement on Epidemic. The simulation result is shown in Figure 6(A).

(1.2) Successful delivery rate under FirstContact router

The simulation result is shown in Figure 6(B). This result shows that, in FirstContact router, the rate of successful delivery is sensitive to the number of selfish nodes, which is related

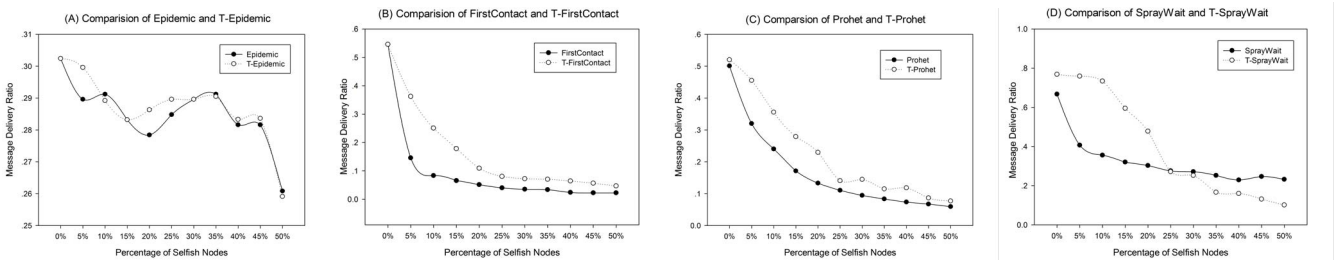


Fig. 6. Rate of successful delivery in four routers.

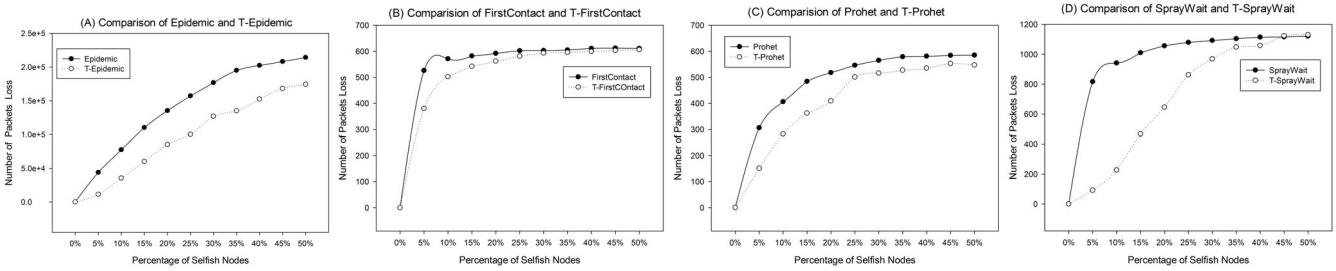


Fig. 7. Number of losing packets in different router.

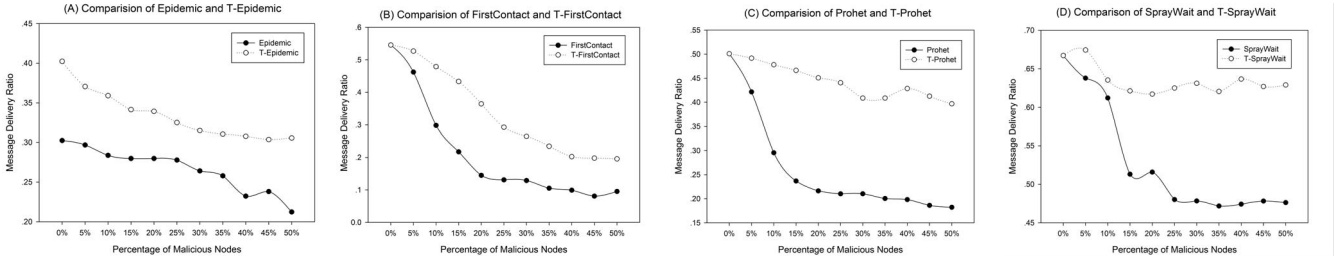


Fig. 8. Rate of successful delivery in four different router.

to that each message only has one copy in FirstContact. While the percentage of selfish nodes is around 10%, the rate of successful delivery has decreased to 0.05. When the trust model is added, the decreasing rate of successful delivery becomes slower. But with the increase of the percentage of selfish nodes, the rate of successful delivery becomes low.

(1.3) Successful delivery rate under Prophet router

The simulation result is shown in Figure 6(C). This result shows that with the trust model, Prophet performs well on resisting selfish nodes. The rate of successful trust delivery is maintained at a high level. Prophet delivers its messages by the probability of meeting with other nodes, thus this router has a high probability of successful delivery. After the trust model is added into Prophet router, packet loss is reduced at some degrees. But single-copy is adopted in the experiments, single-copy is sensitive to packet loss. Thus, the rate of successful delivery is low while the percentage of selfish nodes is large.

(1.4) Successful delivery rate under SprayWait router

In this experiment, we set each node to have 4 copies of messages. The result is shown in Figure 6(D). The result shows that, while the percentage of selfish nodes is less than 20%, the trust model performs well on resisting selfish nodes. It can be explained that SprayWait is based on multi-cpu router, in which trust accumulation is faster than in single-copy router. However, as the selfish nodes become more, each node in this router will send more than one copies of messages. For not receiving ACK immediately, some normal nodes may be considered falsely as selfish nodes, which result in that the rate of successful delivery is less than the result in router without the trust model. Normal nodes need more time to accumulate the trust of other nodes while there are a lot of selfish nodes.

(2) Numbers of packets loss. We compare the number of lost packets when having a trust model with the one without trust model, in order to analyze the trust model more accurately. The result is shown in Figure 7.

This result, after the trust model is added into Epidemic router, is shown in Figure 7(A). This router chooses forwarding strategy according to ACK instead of sending messages by Epidemic method. This method reduces the interaction with selfish nodes, decreasing the number of packets loss. In improved router FirstContact, shown as Figure 7(B), while selfish nodes are with small amount, trust mode can achieve good performance on this indicator of packet loss. In improved Prophet algorithm, packets loss rate decreases, as shown in Figure 7(C). In improved SprayWait router, those nodes with high credit are allowed to forward messages. The impact of trust model is the most obvious in this router, as shown in Figure 7(D).

3) Malicious Nodes: Obviously, malicious nodes have negative impact on OMSN. The malicious behaviors include sending spam messages, dropping received messages, making conspiracy attack and so on. Without an effective trust model, increasing malicious nodes would decrease the rate of successful message delivery. The reason is that malicious nodes will drop packets and make flooding attack. Packet loss has been discussed in the above, we only discuss spam messages in this part. Malicious nodes produce a large amount of spam and inject them into the networks. Thus, many normal nodes may be occupied by these useless messages, which cause them to unable to carry and forward normal messages immediately, decreasing the rate of successful message delivery of the entire network. The comparisons of impacts on unimproved and improved protocols are shown in Figure 8.

The experiment result shows that, malicious nodes have big negative impacts on the network. After spam are sent or messages are dropped by misbehaving nodes, normal nodes can do nothing but drop those normal messages they carried. Thus, the rate of successful delivery decrease as the number of malicious nodes increases. The trust model shows good performance with different routers, especially with FirstContact and Prophet, shown as Figure 8(B) and 8(C). The reason is that both routers belong to single-copy routers, which mean there is only one copy in network. In a router with the trust model, nodes decide whether accept request of receiving messages according to neighbor nodes' fitness condition. If it has a low fitness value, the request would be refused to avoid receiving garbage messages, which improves the successful forwarding rate.

Malicious nodes in network spread spam, which cost network resources. Thus, network load can be considered as an important factor of evaluating the effect of malicious modes. Changes of routers in network loads are shown in Figure 9.

Based on experiment results, we conclude that, after adopting our trust model, normal nodes can identify most of malicious nodes, so that they can avoid to receive and forward messages sent by those misbehaving nodes. Therefore it can save network resources. Our trust model shows good performance in multi-copies routing protocols. We combine those protocols with our trust model, and it makes messages forwarding process be risk perception. It enhances the ratio of successful messages delivery.

4) Analysis of Detection: In this section, we analyze the ratio of detecting misbehaving nodes in our simulation. we calculate the average precision rate and recall rate in detection of misbehaving nodes in the entire network. Detection result of packet loss rate and spam are shown in Table VI and VII.

The result shows that, in a network with few selfish nodes, the precision rate of detecting selfish nodes' packet loss behaviour is low, but the recall rate is bigger than 0.2. With the ratio of selfish nodes becoming bigger, the precision rate of detecting packet loss behaviour is becoming higher. But the recall rate decrease to 0.12 and the value of F-score also decreases. All in all, while the ratio of selfish nodes is 15%, the network has the best performance in detecting packet loss behaviour.

TABLE VI
DETECTION OF PACKETS LOSS

Ratio of selfish nodes	Average value of accuracy rate	Average value of recall rate	Average value of F
5%	0.44	0.228	0.300
10%	0.553	0.267	0.360
15%	0.534	0.222	0.314
20%	0.54	0.171	0.259
25%	0.564	0.175	0.267
30%	0.612	0.147	0.237
40%	0.642	0.129	0.215
50%	0.681	0.122	0.207

TABLE VII
DETECTION OF FLOODING BEHAVIOR

Ratio of selfish nodes	Average value of accuracy rate	Average value of recall rate	Average value of F
5%	0.489	0.166	0.247
10%	0.556	0.131	0.212
15%	0.794	0.261	0.393
20%	0.815	0.236	0.366
25%	0.769	0.223	0.345
30%	0.799	0.208	0.330
40%	0.794	0.276	0.409
50%	0.837	0.253	0.389

The result shows that the precision rate of identifying abnormal nodes drops as the number of abnormal nodes increases, because the number of nodes with value less than the threshold increases. The trust model will assign low trust degree values to abnormal nodes because of their bad behaviour. In the experiment of conspiracy attack, we set up 90 normal nodes, and the number of conspiracy nodes changes from 2 to 16. For nodes are in distributed environment and in lack of unified certificate center, we use the precision rate and recall rate of normal nodes to measure the capability of detecting all conspiracy nodes in the network. Normal nodes' precision rate and recall rate of considering conspiracy are shown in Table VIII.

Also, the accuracy of each node's judge on conspiracy nodes increases as pairs of conspiracy nodes increasing. Recall rate is stable and is around 0.1 or a little bit higher, which means each node can recognize 10% conspiracy nodes in average. For the network with 8 pairs of conspiracy nodes, the average detection rate has a higher F value. Recognition rate increases as the trust degree record passing by and accumulated in the trust degree list.

We do not allow normal nodes sending null message after 12 hours. While simulation is done in less than 14 hours, recall rate and precision rate of conspiracy attack keep rising. After 14 hours, recall rate and precision rate of conspiracy attack keep unchanged. Precision rate is stable at 0.42, recall rate is stable at 0.15. It can be explained that after 12 hours, node collect most nodes' ACK message, and can judge that whether the nodes of conspiracy nodes are in a pair. With the ACK message added into its list, normal node's precision rate of judging conspiracy nodes will come into convergence.

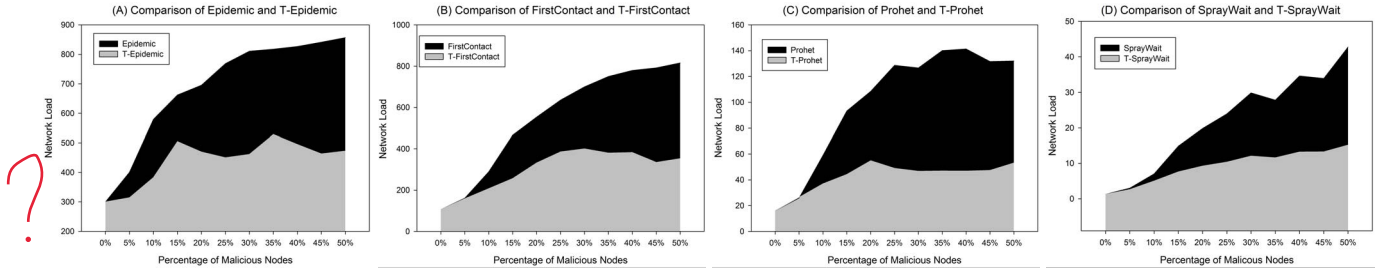


Fig. 9. Rate of network load in four routers.

TABLE VIII
CONSPIRACY ATTACKS DETECTION

Pair of CA	Average detection ratio	Average recall ratio	Average ratio of F value
2	0.100	0.106	0.103
4	0.252	0.150	0.188
6	0.301	0.124	0.176
8	0.413	0.148	0.217
10	0.455	0.125	0.196
12	0.475	0.124	0.197
14	0.468	0.117	0.187
16	0.539	0.111	0.184

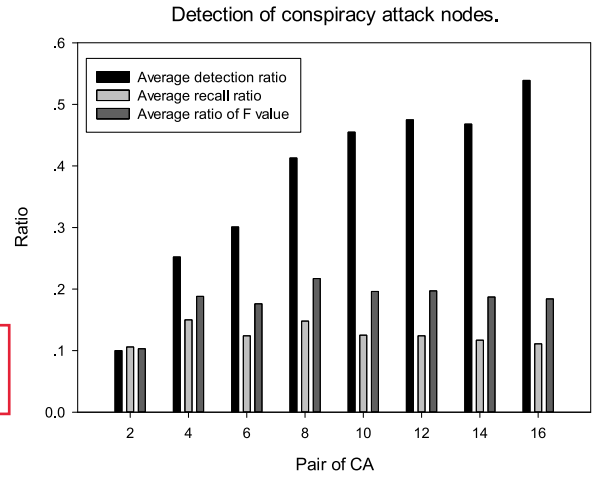


Fig. 10. Detection of conspiracy attack nodes.

Conspiracy attack is one of malicious attacks, and it's difficult to be detected. If some nodes are always paired with each other to give good evaluation by ack, the behavior can be considered to be risk of conspiracy attack. According to the 2-Hop Ack scheme in our model, malicious nodes will drop messages by using conspiracy method, and generate feedback for confirming. For example, if a malicious node m_1 receive the Msg sent by a normal node S, then m_1 drop the Msg, m_1's partner m_2 then sends a fake feedback to S, making S believe that Msg has been delivered from m_1 to m_2. Finally, S will enhance the trust to m_1.

For verification of our model to tackle the conspiracy attack, we set the number of normal nodes in the network to be 90, the number of pairs of conspiracy attack(CA) nodes vary from 2 to 16. Because it is in a distributed environment, we use the average precision ration and recall ratio of detection of conspiracy attack. The detection ratio and recall ratio are shown in the Figure 10.

V. CONCLUSION

In this paper we propose a trust framework for OMSN. In the framework, a series of models and algorithms are designed for current routing protocols. The experiment results of this paper are encouraging, but it can be regarded as the first step towards an effective trust model for OMSN. Improving the effectiveness and performance of the proposed trust model would be our next step. For the calculation of connectivity, fitness, and satisfaction, the approach is still primitive. It would be exciting to find out if there are better methods to capture these three factors.

ACKNOWLEDGMENT

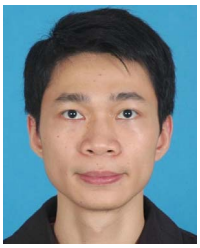
The authors thank the reviewers for their comments.

REFERENCES

[1] M. Duggan, *Photo and Video Sharing Grow Online*, Pew Internet, Washington, DC, USA, 2013.
[2] V. Arnaboldi, M. Conti, and F. Delmastro, "Implementation of CAMEO: A context-aware middleware for opportunistic mobile social networks" in *Proc. IEEE Int. Symp. World Wireless Mobile Multimedia Netw. (WoWMoM)*, Lucca, Italy, 2011, pp. 1–3.
[3] R. Ranjan, N. K. Singh, and A. Singh, "Security issues of black hole attacks in MANET," in *Proc. Int. Conf. Comput. Commun. Autom.*, Noida, India, 2015, pp. 452–457.
[4] Y. Li and J. C. S. Lui, "Epidemic attacks in network-coding-enabled wireless mesh networks: Detection, identification, and evaluation," *IEEE Trans. Mobile Comput.*, vol. 12, no. 11, pp. 2219–2232, Nov. 2013.
[5] M. Patel and S. Sharma, "Detection of malicious attack in MANET a behavioral approach," in *Proc. 3rd IEEE Int. Adv. Comput. Conf. (IACC)*, Ghaziabad, India, 2013, pp. 388–393.
[6] T. Beth, M. Borcherding, and B. Klein, "Valuation of trust in open networks," in *Proc. Eur. Symp. Res. Comput. Security (ESORICS)*, vol. 875. Brighton, U.K., 1994, pp. 3–18.
[7] R. Yahalom, B. Klein, and B. Thomas, "Trust-based navigation in distributed systems," *Comput. Syst.*, vol. 7, no. 1, pp. 45–73, 1994.
[8] B. Yu and M. P. Singh, "A social mechanism of reputation management in electronic communities," in *Cooperative Information Agents IV—The Future of Information Agents in Cyberspace*. Heidelberg, Germany: Springer 2000, pp. 154–165.
[9] M. Venkatraman, B. Yu, and M. P. Singh, "Trust and reputation management in a small-world network," *Proc. 4th Int. Conf. MultiAgent Syst.*, 2000, Boston, MA, USA, 2000, pp. 449–450.
[10] J. Caverlee, L. Liu, and S. Webb, "Socialtrust: Tamper-resilient trust establishment in online communities," in *Proc. 8th ACM/IEEE CS Joint Conf. Digit. Libraries*, Pittsburgh, PA, USA, 2008, pp. 104–114.

[11] J. Caverlee *et al.*, "SocialTrust++: Building community-based trust in social information systems," in *Proc. 6th Int. Conf. Collaborative Comput. Netw. Appl. Worksharing (CollaborateCom)*, Chicago, IL, USA, 2010, pp. 1–7.
[12] L. Xiong and L. Liu, "PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 7, pp. 843–857, Jul. 2004.
[13] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in P2P networks," in *Proc. 12th Int. Conf. World Wide Web (WWW)*, Budapest, Hungary, 2003, pp. 640–651.
[14] R. Zhou and K. Hwang, "PowerTrust: A robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 4, pp. 460–473, Apr. 2007.
[15] A. Josang and J. Haller, "Dirichlet reputation systems," in *Proc. 2nd Int. Conf. Availability Rel. Security (ARES)*, Vienna, Austria, 2007, pp. 112–119.
[16] T. Chun-Qi, Z. Shi-Hong, T. Hui-Rong, W. Wen-Dong, and C. Shi-Duan, "A new trust model based on reputation and risk evaluation for P2P networks," *J. Electron. Inf. Technol.*, vol. 29, no. 7, pp. 1628–1632, 2007.
[17] J.-L. Hu, Q.-Y. Wu, B. Zhou, and J. H. Liu, "Robust feedback credibility-based distributed P2P trust model," *J. Softw.*, vol. 20, no. 10, pp. 2885–2898, 2009.
[18] F. Li, J. Wu, and A. Srinivasan, "Thwarting blackhole attacks in disruption-tolerant networks using encounter tickets," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, 2009, pp. 2428–2436.
[19] Y. Liu and K. Wang, "Trust control in heterogeneous networks for Internet of Things," in *Proc. Int. Conf. Comput. Appl. Syst. Model. (ICCASM)*, Taiyuan, China, 2010, pp. V1-632–V1-636.
[20] Y. B. Saied, A. Olivereau, D. Zeghlache, and M. Laurent, "Trust management system design for the Internet of Things: A context-aware and multi-service approach," *Comput. Security*, vol. 39, pp. 351–365, Nov. 2013.
[21] D. Chen *et al.* "TRM-IoT: A trust management model based on fuzzy reputation for Internet of Things," *Comput. Sci. Inf. Syst.*, vol. 8, no. 4, pp. 1207–1228, 2011.
[22] P. N. Mahalle, P. A. Thakre, N. R. Prasad, and R. Prasad, "A fuzzy approach to trust based access control in Internet of Things," in *Proc. Wireless Commun. Context Aware Multi Service Trust Manag. Syst. Veh. Technol. Inf. Aerosp. Electron. Syst. (VITAE)*, Atlantic City, NJ, USA, 2013, pp. 1–5.
[23] M. Nitti, R. Girau, and L. Atzori, "Trustworthiness management in the social Internet of Things," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1253–1266, May 2014.
[24] X. Meng, G. Geng, and J. Ma, "A comprehensive trust evaluation model for social networks," *J. Comput. Theor. Nanosci.*, vol. 13, no. 2, pp. 1330–1336, 2016.
[25] L. Yao, Y. Man, Z. Huang, J. Deng, and X. Wang, "Secure routing based on social similarity in opportunistic networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 594–605, Jan. 2016.
[26] M. K. Denko, T. Sun, and S. I. Woungang, "Trust management in ubiquitous computing: A Bayesian approach," *Comput. Commun.*, vol. 34, no. 3, pp. 398–406, 2011.
[27] I.-R. Chen, F. Bao, M. J. Chang, and J.-H. Cho, "Dynamic trust management for delay tolerant networks and its application to secure routing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 5, pp. 1200–1210, May 2014.
[28] A. Keränen, T. Kärkkäinen, and J. Ott, "Simulating mobility and DTNs with the ONE," *J. Commun.*, vol. 5, no. 2, pp. 92–105, 2010.

Eric Ke Wang received the Ph.D. degree from the Department of Computer Science, University of Hong Kong in 2009. He is an Associate Professor with the Harbin Institute of Technology, Shenzhen, China, where he is currently a Senior Researcher with the Key Laboratory of Shenzhen Internet Information Collaborative Technology and Application. His main research interests include network security and deep learning. He received two granted projects from National Science Funding of China. He has developed two software platforms for opportunistic social networks and obtained two authorized related patents.



Yueping Li received the Ph.D. degree in computer science from Sun Yat-sen University in 2008. He is currently an Associate Professor of Shenzhen Polytechnic. His research interests involve Web mining, graph algorithm, and optimization.

Yunming Ye received the Ph.D. degree from Shanghai Jiao Tong University in 2004. He is a Professor with the Department of Computer Science, Shenzhen Graduate School, Harbin Institute of Technology (HIT). He is the Director of the Shenzhen Key Laboratory of Internet Information Collaboration and the Director of Intelligent Enterprise Computing Research Laboratory, HIT, Shenzhen. He has published over 60 refereed papers in international journals or conferences, including the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, the IEEE TRANSACTIONS ON NEURAL NETWORKS, *Pattern Recognition*, and ACM SIGKDD Conferences on Knowledge Discovery and Data Mining. His research areas are data mining, intelligent Web search, and business intelligence.

S. M. Yiu received the B.Sc. degree in computer science from the Chinese University of Hong Kong, the M.S. degree in computer and information science from Temple University, and the Ph.D. degree in computer science from the University of Hong Kong. He is an Associate Professor with the University of Hong Kong. He has published over 80 refereed papers in international journals or conferences. He was a recipient of two research output prizes, one from the department in 2013 and one from the faculty in 2006.

Lucas C. K. Hui (SM'86) received the B.Sc. and M.Phil. degrees in computer science from the University of Hong Kong and the M.Sc. and Ph.D. degrees in computer science from the University of California, Davis. He is the Founder and the Honorary Director of the Center for Information Security and Cryptography, and concurrently an Associate Professor with the Department of Computer Science, University of Hong Kong. Besides actively publishing research papers, he is also involved in consultation work in security systems and in industrial collaboration projects. He is the Principal Investigator of several applied research projects relating to I.T. security with a total sum of around $2.5 millions. His research interests include information security, authentication services, network security, cryptography, and electronic commerce.